LAUREA

Avoin tietokantapalvelu tieteellisen laskennan asiakkaille

Salonen, Sami

2009 Laurea Kerava

Laurea-ammattikorkeakoulu
Laurea Kerava

Avoin tietokantapalvelu tieteellisen laskennan asiakkaille

Sami Salonen
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Lokakuu, 2009

Laurea-ammattikorkeakoulu
Laurea Kerava
Tietojenkäsittelyn koulutusohjelma
Digitaalinen media

Tiivistelmä

Sami Salonen

Avoin tietokantapalvelu tieteellisen laskennan asiakkaille

Vuosi 2009 Sivumäärä 41

Opinnäytetyössä käydään läpi ja dokumentoidaan projektia, jossa luodaan CSC – Tieteen tietotekniikan keskus Oy:lle avoin tietokantapalvelu, jonka asiakkaina tulevat olemaan tieteellistä laskentaa tekevät tutkijat ja käyttäjät. Projektissa luodaan täysin uusi palvelu, jossa käytetään hyödyksi jo olemassa olevaa CSC:ltä löytyvää infrastruktuuria, käytännön kokemuksia ja tietotaitoa.

Projekti toteutetaan CSC:n Datan ja tiedon palvelut -nimiselle palvelualueelle, joka toimii projektin omistajana. Projekti toteutetaan CSC:n sisäisenä projektina, jossa toimii erillinen ohjausryhmä. Ohjausryhmän tehtävänä on pitää projekti aikataulussa, ohjeistaa projektityöryhmää ja toimia neuvoa-antavana kumppanina projektin edetessä. Ohjausryhmässä toimii kolme henkilöä, joilla on tietoa projektin taustoista ja mahdollisuuksia ohjata projektin etenemistä.

Opinnäytetyön teoreettisessa osassa käydään läpi tietokantojen testausta, käyttöönottoa ja ympäristön suunnittelua. Lähdeaineistona on erityisesti MySQL-tietokantojen käyttöön liittyvää kirjallisuutta ja sähköisiä viitteitä, koska käytössä on paljon uutta teknologiaa joista ei kirjallista materiaalia vielä löydy.

Projektissa käydään läpi tehtyjä uusia ratkaisumalleja, joita voidaan soveltaa muissa projekteissa. Projektin edetessä jouduttiin tekemään monia omia räätälöityjä ratkaisuja, jotta halutut ominaisuudet saatiin upotettua CSC:n ympäristöön.

Asiasanat: tietokannat, MySQL, käyttöjärjestelmät, Linux, benchmarking

Laurea University of Applied Sciences
Laurea Kerava
Business Information Systems
Digital Media

Abstract

Sami Salonen

Open database service for scientific computing

Year 2009 Pages 41

This Bachelor's thesis describes and documents a project in which a new service will be created. CSC – IT center for science will produce a new database service for scientists and researchers where they can store their scientific data produced with CSC's supercomputers. The new service will utilize already existing infrastructure, common practices and know-how.

The project will be created for CSC's Data Services for Science and Culture service area which also maintains the project ownership. A separate steering group for the project will be guiding the project team. The steering group will keep the project on schedule, gives guidance and will sort out CSC's internal issues for the project. Members of the steering group will consist of 2 or 3 people who have enough expertise on databases.

Theoretical part of the thesis consists of database benchmarking, testing, designing and implementation. As referring literature I've used several new and old technical manuals in the field of databases. In addition, also older database designing related material was used as well as new electronic material collected from Internet.

The project will describe new implementations which can be utilized in new projects to come. We created several new ways to solve well known problems with MySQL database so that the desired features could be presented.

Key words: databases, MySQL, operating systems, linux, benchmarking

Sisällys

1 JOHDANTO

Opinnäytetyön aihe on ”Avoin tietokantapalvelu tieteellisen laskennan asiakkaille”. Opinnäytetyö tehdään projektina, jossa rakennetaan tietokantapalvelu CSC - Tieteen tietotekniikan keskus Oy:n suurlaskentapalveluja käyttäville asiakkaille. Opinnäytetyö toimii dokumentointiprojektina varsinaiselle projektille, koska tämä opinnäytetyö tulee olemaan osa projektille tehtävää dokumentaatiota.

Asiakkaille ei tällä hetkellä tarjota mahdollisuutta rakentaa omia tietokantoja joihin voisi tallentaa esimerkiksi tutkimustietoja. Tämä on ollut selkeä puute CSC:n tarjoamassa palvelutarjonnassa. Omien tietokantojen rakentaminen ja ylläpito avaisi asiakkaille monia uusia ja tehokkaita tapoja tutkimustulostensa tiedonhallintaan ja datan monipuoliseen hyödyntämiseen. Samalla tutkimustulosten jatkojalostaminen helpottuu, kun tiedot saadaan yhteen yhtenäiseen muotoon, eli tietokantaan. Tällöin tietoja voidaan jakaa eteenpäin määrämuotoisena tietokantavedoksena, joka on tärkeää tietojen eheyden kannalta.

Sen lisäksi, että CSC pystyy tarjoamaan uuden palvelun asiakkaille, projektin myötä tietokantaosaamisen taso CSC:ssä paranee, mikä hyödyntää mahdollisia jatkoprojekteja tämän palvelun laajentamiseksi useampaa tietokanta-alustaa koskevaksi. Toistaiseksi tämän projektin suosio on CSC:llekin melkoinen arvoitus, sillä vaikka tiedusteluja tällaisesta palvelusta on olemassa, ei sen suosiota osata vielä täysin arvioida. Tarkempaa kartoitusta tietokantapalvelun käytöstä ei ole tehty. Projektin rahoitus on tästä syystä hyvin konservatiivinen, jos käykin niin, että ennakko-odotuksista huolimatta palvelu ei saakaan tarvittavaa kiinnostusta ja asiakaskuntaa.

2 Opinnäytetyön taustat ja tavoitteet

CSC saa ajoittain tiedusteluja asiakkailta suurten tietomäärien tallentamiseen. Usein tietojen tallettamiseen riittää normaali asiakkaille tarjottu levytila, jota CSC:n talletuspalvelujen levypalvelimilta löytyy. Suurimmalle osasta asiakkaista riittää heidän oma dedikoitu levytilansa hyvin. CSC:llä on tarjolla useita muita vaihtoehtoja suurempien tietomäärien lyhyt- ja pitkäaikaiseen tallentamiseen. Esimerkiksi 30 päivän pituiseen tiedon talletukseen löytyy metawrkniminen levyalue, jonne voi tallentaa materiaaliaan lähes rajoituksetta. Archive-nimiselle levyalueelle voi tallentaa oletuksena 550 Gigatavua materiaalia, joka varmistetaan kahdelle erilliselle nauhajärjestelmälle pidemmäksi aikaa. Todella pitkäaikaisesta materiaalin talletuksesta pitää sopia CSC:n kanssa aina erikseen, ettei inhimillisistä syistä johtuvia tietojen katoamisia tai poistoja sattuisi. (CSC – Tieteen Tietotekniikan Keskus 2009.)

Suurten tietomäärien analysoinnissa tarvitaan usein tietokantapohjaisia ratkaisuja, joita CSC on jollain tasolla pystynyt tarjoamaan. Tällä hetkellä CSC tarjoaa noin 70 eri tieteenalojen (fysiikka, kemia, kielitieteet jne.) tietokantaa tutkijoiden käyttöön. Nämä tietokannat ovat kuitenkin hakutietokantoja, joihin voi kohdistaa hakuja, mutta niiden sisältöön ei voi itse vaikuttaa.

Usein tutkijat kohtaavatkin ongelmana sen, että heidän valmiille datalle ei ole järkevää talletuspaikkaa CSC:llä. Ratkaisuna on ollut esimerkiksi oman yliopiston tai tutkimuslaitoksen tietokantapalvelujen käyttäminen mutta tämä aiheuttaa taas turhia tiedonsiirtoja CSC:n ja muiden laitosten välillä. Useasti tietomäärätkin ovat jo niin massiivisia, ettei niiden siirtämiseen ole mielekkäitä vaihtoehtoja. Jos CSC pystyisi tarjoamaan täydellisenä pakettina laskentaa ja siihen sisältyvää tulosten analysointia ja talletusta, olisi CSC parantamassa palvelutasoaan huomattavasti.

Projektin tuloksena syntyy selkeä uusi asiakkaille suunnattu palvelu. Lisäksi projektin myötä tietokantaosaamisen taso CSC:ssä paranee, mikä hyödyntää mahdollisia jatkoprojekteja tämän palvelun laajentamiseksi myös useampaa tietokanta-alustaa koskevaksi. Tietokanta-alusta on MySQL, koska se on ainoa täysin tuettu ympäristö CSC:llä. Tietokanta-alustan valintaan projektiryhmä ei voi vaikuttaa.

2.1 Opinnäytetyön rajaukset

Projektia koskevat seuraavat rajaukset:

- Projektissa keskitytään vain MySQL-tietokanta-alustaan.
- Projekti hyödyntää CSC:n pienpalvelinten ylläpitoa laitteiston asentamisessa ja ylläpidossa.
- Projekti ei tule muuttamaan olemassa olevia CSC:n sisäisiä prosesseja, ilman erillisiä vaatimuksia. Tällaisia prosesseja ovat mm. käyttäjähallinto ja resurssien hallinta.
- Projekti ei tule rakentamaan asiakkaille heidän tietokantoja. Dbadm-prosessi voi tarvittaessa antaa konsultointiapua tällaisissa tapauksissa.

2.2 CSC – Tieteen tietotekniikan keskus Oy

CSC on opetusministeriön omistama ja hallinnoima yhtiö, joka tarjoaa korkeakouluille, tutkimuslaitoksille ja yrityksille tietoteknisiä tukea ja resursseja. CSC:n englanninkielinen nimi on CSC - IT Center for Science, joka on samalla yhtiön toinen virallinen nimi. CSC perustettiin vuonna 1971 ja oli aluksi osa suurempaa kokonaisuutta, joka tunnettiin tuolloin nimellä Valtion tietokonekeskus (VTKK). 1986 Valtion tietokonekeskus jaettiin pienempiin osiin, jolloin CSC:kin myös sai nykyisen nimensä. Tuosta jaosta syntyi muita vielä nykyisinkin tunnettuja yhtiöitä, kuten tällä hetkellä jo useasti nimensä vaihtanut tietotekniikan palveluyritys Tieto.

CSC:n palvelut voidaan jakaa karkeasti viiteen eri alueeseen:

- Laskentapalvelut
- Funet-palvelut
- Ohjelmistopalvelut
- Datan ja tiedon palvelut
- Tietohallintopalvelut

Enimmäkseen ihmiset tunnistavat CSC:n sen tarjoamista supertietokoneista ja niiden laskentatehosta. Tällä hetkellä CSC:n hallinnassa on 2 supertietokonetta (murska.csc.fi ja louhi.csc.fi), jotka ovat vieläkin maailman top 500-supertietokonelistauksessa sijoilla 204 ja 238. Korkeakouluopiskelijoiden ja henkilöstön piirissä CSC:n tarjoama Funet-verkko on tuttu käsite. Verkko kattaa kaikki yliopistot ja ammattikorkeakoulut ja käyttäjiä sillä on tällä hetkellä noin 300 000. Verkolla on asiakkaina tutkimuslaitoksia ja muita valtion omistamia laitoksia, kuten Kansallinen Audiovisuaalinen Arkisto.

CSC:llä oli ensimmäinen supertietokone, joka on hankittu Suomeen. Tuolloinen supertietokone oli Univac 1108 mainframe-kone. Tuolloin tätä suurhankintaa pidettiin akateemisen laskennan pelastajana ja suunnitelmissa oli, että kone olisi turvannut laskentatarpeet seuraavat 30 vuotta. Valitettavasti tuon aikakauden optimistiset unelmat ovat jo aikapäiviä sitten haihtuneet ilmaan, koska laskentatehovaatimukset ovat nousseet räjähdysmäisesti vuodesta toiseen. (CSC – Tieteen Tietotekniikan Keskus 2008.)

Yhtiö on kasvanut jatkuvasti, tosin hallitusti, ja työntekijöiden määrä on ollut jatkuvassa kasvussa joka vuosi. Tällä hetkellä työntekijöitä on noin 160 ja ainoa rajoite kasvulle tuntuu olevan ammattitaitoisen työvoiman saanti ja vapaiden toimistotilojen saaminen. Suurin osa yhtiön rahoituksesta tulee opetusministeriöltä, mutta esimerkiksi EU-rahoituksen määrä on viime vuosina kasvanut huomattavasti.

3 INFRASTRUKTUURI

CSC:n oma sisäinen infrastruktuuri sanelee joitain reunaehtoja projektille, joihin projektiryhmä ei voi vaikuttaa. Tulevan tietokantapalvelun käyttäjän on oltava CSC:n laskentakoneympäristön käyttäjä, että hän voi käyttää tarjoamaamme palvelua. Käyttäjän on oltava mukana laskentaprojektissa, jolle on luotu tarvittavat tunnukset ja annettu laskenta-aikaa supertietokoneympäristöön. Lähtökohtaisesti kaikki CSC:n asiakkaan ovat saaneet käyttäjätunnukset CSC:n laskentaympäristöön, mutta poikkeustapauksiakin voi löytyä. Käyttäjätunnusten hallinnointi on keskitetty Usermgr-prosessille, joka käytännössä hoitaa kaikki CSC:n asiakastunnusten käsittelyt. Usermgr-prosessin sisällyttäminen projektiimme tulee olemaan yksi suurimmista ei-teknisistä haasteista, joita joudumme ratkaisemaan.

Tietokantayhteyksien muodostamista palveluun joudutaan rajoittamaan siten, että vain CSC:n omista asiakaskäyttöisistä laskentapalvelimista on mahdollista ottaa yhteys tietokantapalvelimeen. Tämä on tehty siksi, että mahdolliset tietoturvaa uhkaavat väärinkäytökset saadaan minimoitua. On todella harvinaista, että tietokantapalvelimet olisivat täysin avoinna Internetissä, jolloin ne ovat todella suuressa vaarassa joutua erilaisten hyökkäysten kohteiksi. Näin suurta riskiä omassa projektissamme ei haluttu ottaa, koska tällaisen menettelyn riskit ovat huomattavasti suuremmat kuin sillä saavutettavat edut. Asiakkaiden on tosin mahdollista käsitellä tietokantojaan suoraan kotikoneiltaan SSH-tunnelointitekniikan avulla. Tällöin luodaan tunneli kotikoneelta CSC:n laskentaympäristön läpi tietokantakoneelle tällä hetkellä hyvin tietoturvallisen SSH-protokollan avulla, jolloin tietokannan kotikäyttö on periaatteessa mahdollista. Tätä käytäntöä ei tulla kuitenkaan aktiivisesti mainostamaan, koska pyrimme siihen että yhteydenotot tietokantaan tapahtuvat CSC:n laskentaympäristöstä. Tällöin voimme olla varmoja, ettei yhteensopivuusongelmia asiakasohjelmistojen kanssa tule, kun saamme itse päivittää ja määrätä omasta ympäristöstämme. Iso osa helpdesk-palvelumme ongelmista on juuri tällaisia yhteensopivuusongelmia, joita eri ohjelmistojen versioiden suuri määrä aiheuttaa.

3.1 Sisäiset yhteistyöryhmät

Tietokantojen talletukseen tulemme käyttämään CSC:n sisäisen Tallennuspalvelut-nimisen ryhmän tarjoamia palveluja. Tallennuspalvelut-ryhmä huolehtii laskennallisen tieteen ja kansallisten yhteistyökumppanien tiedon tallennukseen tarvitsemien tietokoneresurssien ja palvelujen kehittämisestä ja ylläpidosta.

Datan tallennuspalvelut huolehtivat seuraavista toiminnoista:

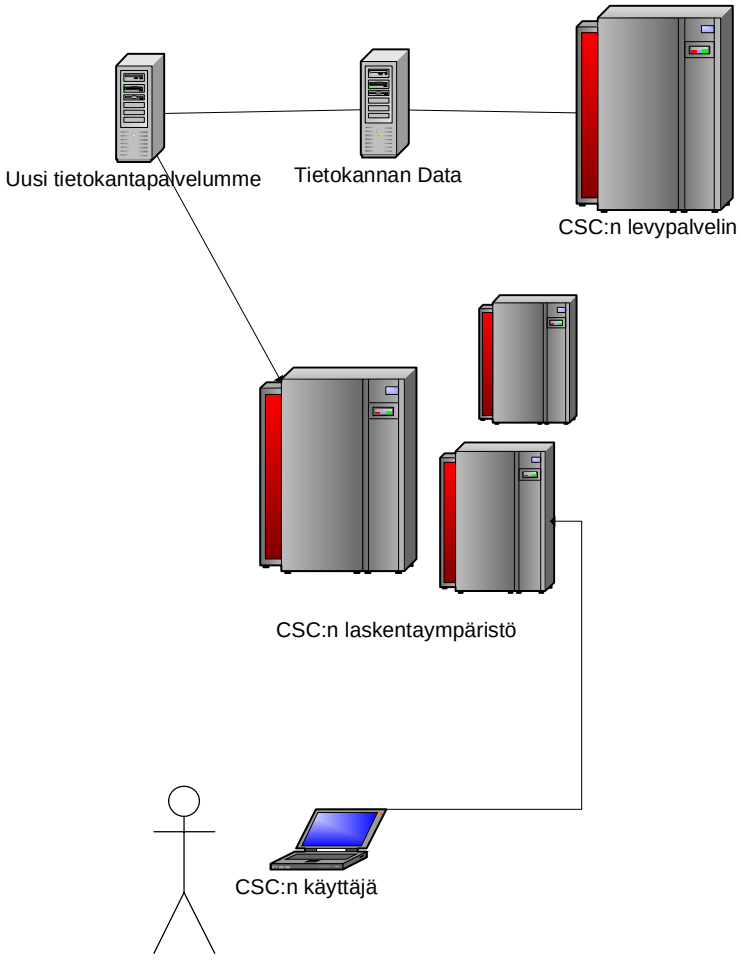
- Levypalvelimien kehitystehtävät ja operatiivisen ylläpidon sekä levypinnan jakamisen CSC:n datan tallennuspolitiikan mukaisesti.
- Arkistopalvelimen kehitystehtävät ja operatiivisen ylläpidon sekä arkistotilan jakamisen CSC:n datan tallennuspolitiikan mukaisesti.
- Varmuuskopiointijärjestelmän kehitystehtävät ja operatiivisen ylläpidon sekä tiedostojen palauttamisen tarvittaessa.
- SAN-verkon (Storage Area Network) kehitystehtävät ja operatiivisen ylläpidon
- Datan tallennukseen liittyvät konsultointi- ja projektitehtävät CSC:n omissa kehityshankkeissa sekä sopimuksen mukaan CSC:n asiakkaille

Palvelun kriittisyys ja vaikuttavuus ovat osittain verrattavissa tietoliikenneverkon toimintavaatimuksiin. Erityisesti levypalvelinjärjestelyiltä edellytetään korkean käytettävyyden (High availability) kriteerien täyttymistä, sillä CSC:n muiden palvelujen saatavuus riippuu olennaisesti levypalvelimien toiminnasta. Datan säilyvyys-, eheys- ja saatavuusvaatimukset ovat 100 %.

Datan tallennuspalvelujen loppukäyttäjiä ovat korkeakoulujen ja sopimusasiakkaiden tutkijat, CSC:n henkilöstö sekä osin korkeakoulujen ja opetusministeriön henkilökunta (CSC – Tieteen Tietotekniikan Keskus 2009).

Itse MySQL-tietokannat tullaan tallettamaan Tallennuspalvelut-ryhmän tarjoamalle levypinnalle, jota käytetään sisäisen SAN-verkon yli (kuva 1). Tällä hetkellä arvioitu tallennusmäärä on noin kaksi Teratavua, mutta tämä määrä on tällä hetkellä minimivaatimus. Tallennustilaa voidaan lisätä, jos palvelu sitä vaatii eikä nykyinen levytila riitä.

CSC:n pienpalvelimista, joita tekemässämme projektissa tullaan käyttämään, vastaa Palvelutuotantoalustat-niminen ryhmä. Ryhmä hoitaa itse laitteiden hankinnan, käyttöjärjestelmien perusasennukset ja laitteiden asentamisen verkkoon. Käyttöjärjestelmien tietoturva on hoidettu myös saman ryhmän kautta. Tietoturvapäivitykset, joita kaikki käyttöjärjestelmät ja ohjelmistot tuntuvat tarvitsevat, tulevat keskitetysti Palvelutuotantoalustat-ryhmältä.



Kuva 1: Kaaviokuva tietokantapalvelun suunnitellusta laitteistoinfrastruktuurista

3.2 MySQL-tietokanta

MySQL on todella suosittu ja tehokas tietokantamoottori, joka on todella laajassa käytössä halvan hintansa takia. Peruskäytössä MySQL-tietokanta on ilmainen, jolloin sen hankinnasta ei koidu lainkaan kustannuksia. MySQL on alun perin ruotsalainen yritys, jonka Sun Microsystems osti vuoden 2008 tammikuussa. Uudesta kaupallisesta omistajastaan huolimatta MySQL on ainakin toistaiseksi täysin ilmainen ja vapaasti käytettävissä useimpiin tarkoituksiin.

MySQL on erityisen suosittu erilaisten web-palveluiden tietokantana. Se on nopea, toimiva ja ylläpitäjien kannalta jopa helposti ylläpidettävä tietokanta. Ylläpitäjiltä tosin odotetaan tiettyjen perusasioiden osaamista sujuvan käytön osalta. MySQL:lle on rakennettu useita eri rajapintoja eri ohjelmointikielille. Suosituimpia ovat varmasti PHP, Perl, Python ja Java. (Wikipedia 2008.)

Tietokantojen vertailussa MySQL on saanut aiemmin moitteita, kun sitä on verrattu esimerkiksi Oraclen tai Sybasen kaltaisiin järeämpiin tietokantoihin, koska siitä on puuttunut esimerkiksi aidot vierasavaimet (foreign keys) ja transaktiot (transactions). MySQL:ään lisätyn InnoDB-moottorin avulla nämä ongelmat on kuitenkin saatu ohitettua ja MySQL on saanut tunnustusta isompien tietokantojen sarjassa. (Wikipedia 2008.)

Toinen kompastuskivi MySQL:n tiellä on ollut tukipalvelujen saatavuus. Kaupallista tukea myös Suomessa on toki ollut saatavilla mutta se on edellyttänyt melkeinpä suoraa sopimusta MySQL AB:n kanssa. Tämä on pienentänyt suuresti isojen tietojärjestelmien toimittajien kiinnostusta tuotteeseen, koska MySQL ei voi taata täydellistä tukea myymälleen tuotteelle.

Vuoden 2008 tammikuussa Sun Microsystems Inc. hankki MySQL-tuotteen ja MySQL-yhtiönimen itselleen (Sun Microsystems Inc 2008). Tämä tarkoittaa sitä, että kaikki kehitys ja tukitoiminta siirtyy Sun Microsystemsin alaisuuteen. Periaatteessa tämä on tukitoimintojen kannalta todella hyvä uutinen, sillä Sunin tukitoiminta on ollut perinteisesti hyvää. Tosin se on melkoisen kallista, joka sotii tietyssä mielessä Open Source-periaatetta vastaan, mutta joillekin MySQL:ää käyttäville organisaatioille tuen saamisen takaaminen on todella hyvä uutinen. Tukipalvelujen saatavuuden takaaminen lisää varmasti isompien yhtiöiden mielenkiintoa heidän päättäessään tietokantaratkaisuistaan.

Sunin MySQL hankinta tuli kaikille hieman yllätyksenä eikä se saanut kaikkien osapuolten täysin varauksetonta hyväksyntää. Vuoden 2008 aikana MySQL-kehitysryhmästä on kaikonnut todella tärkeitä henkilöitä, jotka ovat olleet mukana kehitystyössä vuosikausia. Tällaisten ydinhenkilöiden korvaaminen uusilla, tuoreilla henkilöillä voi olla positiivinenkin piristysruiske tuotteen kehitykselle, mutta yleisesti on arvioitu, että ydinkehittäjien kaikkoaminen on melkoinen takaisku koko MySQL-tietokannalle. (It-viikko 2009.)

3.3 CSC:n MySQL-tukiprosessi

Dbadm, lyhenne termistä Database administration, on CSC:llä toimiva prosessi, jonka pääasiallisena tehtävänä on hoitaa tietokantapalvelujen toimivuus CSC:llä. Lisäksi Dbadm-prosessin tulee tarvittaessa toimia asiantuntijatehtävissä CSC:n sisäisissä tehtävissä, muissa prosesseissa sekä projekteissa. Tällä hetkellä Dbadm-tukiprosessissa on mukana kolme CSC:n asiantuntijaa. Vastuut prosessissa on jaettu lähinnä omien henkilökohtaisten tietojen ja taitojen mukaan, mutta esimerkiksi poissaolojen ja lomien takia on tärkeää, että kaikki prosessiin osallistuvat osaavat tehdä tärkeimmät tuki- ja ylläpitotoimenpiteet.

Käytännössä dbadm-prosessin tulee huolehtia:

- MySQL-tietokantapalvelimien operoinnista ja konfiguroinnista
- MySQL-tietokantapalvelimien käytettävyyden ja tietoturvan valvomisesta
- MySQL-ympäristön kehityksestä, ylläpidosta ja toteutuksesta
- MySQL-ympäristön valvontatyökalujen kehityksestä, ylläpidosta ja toteutuksesta
- Auttaa uusien tietokantasovellusten suunnittelussa ja käyttöönotossa
- Dbadm-postilistan postien käsittelystä ja posteihin vastaamisesta
- Ohjeiden ja lausuntojen kirjoittamisesta tarvittaessa
- Palvelin-/palvelutietojen ja dokumentaation pitämisestä päivitettynä

Olen itse mukana tässä prosessissa ja sen tietoja ja taitoja tullaan käyttämään hyväksi tässä projektissa. Ylläpitoprosessin määrittelyssä on tehty päätös, että MySQL-tietokantoja asennetaan vain Linux- tai muihin Unix-pohjaisiin käyttöjärjestelmiin. Tämä rajaa myös selvästi meidän vaihtoehtoja hankittavan laitteiston ja projektissa käytettävän käyttöjärjestelmän suhteen. Windows-käyttöjärjestelmäympäristöön MySQL:n ei ole katsottu olevan tarpeeksi luotettavalla pohjalla, joten emme ole harkitsemassa sen käyttöönottoa. Windows-alustalla käytössämme on SQL-Server 2005, joka soveltuu mielestämme paremmin Windows-ympäristössä käytettäväksi.

4 LAITTEISTOALUSTAN VALINTA

Tuleva tietokantapalvelu tulee palvelemaan mahdollisesti suurta asiakaskuntaa, joten sen suorituskyvystä täytyy olla konkreettisia testituloksia. CSC:llä on ollut käytössään jo useampia tietokantapalvelimia, mutta niiden suorituskyky on auttamattomasti kehityksestä jäljessä. Näin ollen niitä ei voida käyttää uuden palvelun ja palvelimen käyttöönotossa.

Projektilla oli käytössää rajallinen budjetti, joka käytännössä rajoitti laitteistoalustan PC-tekniikkaan perustuvaan palvelimeen. Projektin aikana oli mahdollisuus testata myös Sun Microsystemsin Solaris 10 -käyttöjärjestelmällä varustettua laitteistoa. Tähän laitteistoon ei kuitenkaan ollut realistista mahdollisuutta sijoittaa budjetoitua rahaa, koska hinta oli moninkertainen verrattuna PC-tekniikkaan. Myös Dbadm-prosessiin kuuluvien henkilöiden rajalliset tiedot Solaris-käyttöjärjestelmästä vaikuttivat tähän hylkäykseen. Jos olisimme päätyneet Solarikseen, olisimme olleet täysin riippuvaisia CSC:n organisaation muista organisaatioryhmistä, joilla on enemmän osaamista Solaris-käyttöjärjestelmästä.

PC-tekniikalla toteutetut palvelimet ovat CSC:llä varustettu yleensä Windows 2003 tai Red Hat Enterprise Linux käyttöjärjestelmällä. Koska meillä ei ole lainkaan kokemusta Windows-käyttöjärjestelmän sopivuudesta MySQL-käyttöön, niin olimme jo ennalta valinneet käyttöjärjestelmäksi Linuxin. Tämä on osoittautunut omassa käytössämme vakaaksi ja suorituskyvyltään toimivaksi yhdistelmäksi.

4.1 Laitteistojen testimetodit

Laitteistojen testaaminen ja benchmarkkaaminen on monen erilaisen komponentin summa. Kaikki vaikuttaa lähestulkoon kaikkeen. Yksinkertaisella testillä saat vastaukset esimerkiksi seuraaviin kysymyksiin:

- Mitä tapahtuu, jos muutan tietokannan rivien määrän kymmenkertaiseksi? Ovatko tietokantakyselyni vielä tällöin nopeita?
- Auttaako muistin lisääminen? Jos auttaa, niin kuinka paljon?
- Onko uusi palvelin todella kaksi kertaa nopeampi kuin vanha?
- Mitä tapahtuu, jos poistan kyselyjen välimuistin käytöstä?
- Kumpi on nopeampi metodi: yksi alikysely vai kaksi lyhyempää kyselyä?
- Mitä tapahtuu, kun kysely ajetaan useaan kertaan tai ajetaan muiden kyselyiden rinnalla? (Zawodny & Balling 2004, 46)

Testeissä päätimme käyttää koneiden vertailuun tietokantaa rasittavaa ohjelmistoa, jolla voimme suoraan asettaa eri laitteistot vertailukelpoiseen järjestykseen.
Laitteistot päätettiin testata MySQL Benchmark Suitella, joka tulee mukana MySQL-tietokantaohjelmiston jakelupaketissa. Ohjelmisto on suunniteltu testaamaan laitteiston suorituskykyä erilaisin tietokantaan kohdistuvin rasitustestein. Testiohjelmisto antaa siis käytännön tietoa, miten laitteisto toimii. Ohjelmisto on kirjoitettu Perl-ohjelmointikielellä ja tarvitsee toimiakseen myös Perlin erityisen DBI-modulin.

Perl on ohjelmointikieli, jolla saa valmista jälkeä nopeasti aikaan. Kärjistäen voidaan sanoa, että Perl mahdollistaa helppojen töiden tekemisen vielä helpommin ja tekee mahdottomista töistä mahdollisen. Helpot työt ovat päivittäisiä rutiininomaisia töitä, joita voivat olla esimerkiksi raporttien laadinnat tietokantakyselyiden perusteella. Vaikeat työt taas vaativat esimerkiksi useamman eri tietolähteen yhdistämisen, muokkauksen ja kokoamisen, jolloin Perl on todella kätevä työväline oikeissa ja taitavissa käsissä. (Wall , Christiansen & Schwartz 1996, 9).

Testauksessa ei otettu huomioon yksittäisten koneiden erityisiä piirteitä, kuten muistin määrää tai prosessorin tehoa vaan luotimme testausohjelmiston antamaan lopputulokseen. Ohjelmisto otti huomioon kaikki edellä mainitut koneiden vaihtelevat attribuutit ja muuttujat. Loppujen lopuksi olimme kiinnostuneita siitä, miten laitteisto hoitaa työtaakkansa ja siinä suorituskyky oli tärkein kriteeri.

4.2 Testatut laitteistot

Käytössämme oli testilaitteistoja, jotka edustavat hyvin kattavasti tällä hetkellä saatavilla olevaa pienpalvelimissa käytettävää laitteistoalustajakaumaa. Meillä oli testeissä mukana neljä VMware ESX-Server-ohjelmistolla virtualisoitua laitetta ja kaksi Hewlett-Packardin Blade-teknologiaan perustuvaa pienpalvelinta.

Virtualisoidut VMware-pienpalvelimet ovat ylläpitohenkilökunnalle helppoja asentaa ja ylläpitää, koska kaikki palvelinten komponentit voidaan lisätä ja poistaa virtuaalisesti. Juuri tämän laitteiston ylläpidon helppouden takia, ovat virtualisoidut palvelinympäristöt saamassa suurta osuutta markkinoista. Myös CSC:llä muutetaan tällä hetkellä fyysisiä palvelimia virtuaalisiin ympäristöihin, koska tämä kaikki helpottaa ylläpidon työtä, säästää sähköä, selkeyttää infrastruktuuria ja koko laitteiston kokonaiskuva on kaikille selvemmin hahmoteltavissa. Käytössä ei siis ole fyysisiä palvelimia vaan ohjelmistolla simuloitu käyttöjärjestelmä ja laite, jotka näkyvät loppukäyttäjille täysin normaaleina pienpalvelimina. Virtuaalikoneiden isäntäjärjestelmiltä tosin vaaditaan todella rautaista suoritustehoa, että ne pystyvät hoitamaan esimerkiksi meidän tapauksessamme vaativat tietokantamoottorin tekemät kyselyt.

Käytössä olivat seuraavat testilaitteistot:

Yhden prosessorin virtuaalikone

- 1 Prosessori (virtuaalinen) 2GHz
- 4 Gigatavua muistia
- Virtualisoidut kovalevyt (Ext3-tiedostojärjestelmä)

Kahden prosessorin virtuaalikone

- 2 Prosessoria (virtuaaliset) 2GHz
- 4 Gigatavua muistia
- Virtualisoidut kovalevyt (Ext3-tiedostojärjestelmä)

Kahden prosessorin virtuaalikone (8Gb muistia)

- 2 Prosessoria (virtuaaliset) 2GHz
- 8 Gigatavua muistia
- Virtualisoidut kovalevyt (Ext3-tiedostojärjestelmä)

Xeon-prosessoriin perustuva Hewlett-Packardin BL460C-palvelin

- 2 Prosessoria, Intel 5430 Quadcore 2.33 GHz
- 10 Gigatavua muistia
- Paikalliset kovalevyt (Ext3-tiedostojärjestelmä)

Opteron-prosessoriin perustuva palvelin

- Hewlett-Packardin Malli BL465
- 2 Prosessoria, AMD Opteron 2.8 GHz
- 10 Gigatavua muistia
- Paikalliset SAS-kovalevyt (Ext3-tiedostojärjestelmä)

Dedikoitu virtuaalikone

- 2 Prosessoria, Intel Xeon X5355 2.66 GHz
- 8 Gigatavua muistia
- Virtualisoidut kovalevyt (Ext3-tiedostojärjestelmä)

4.3 Testausohjelmisto

Testausohjelmisto on valmis paketti, joka tulee aina mukana MySQL:n lähdekoodipaketissa. Ohjelmistoa ei tarvitse kääntää erillisellä kääntäjällä vaan se on suoraan käytössä, jos testattavasta palvelimesta löytyvät Perl-ohjelmointikieli ja tarvittavat DBI-modulit.

Testiohjelmistoa voi käyttää myös muiden tietokantomoottoreiden testaamiseen. Dokumentoinnin mukaan testejä voi ajaa ainakin PostgreSQL-, Solid- ja mSQL-tietokannoissa. Tästä on erityistä apua, jos valittavana on useampia tietokantamoottoreita ja joutuu tekemään valintoja niiden väliltä. Testiohjelmiston koodi on hyvin yleisluontoista Perl-ohjelmointikieltä, jonka muokkaaminen onnistuu hieman ohjelmointia osaavalta helpohkosti. Tarvittaessa ohjelmistoon voi lisätä minkä tahansa tietokantatuen (Oracle, Sybase, Informix, DB2), johon löytyy tarvittava DBI-ajuri. (Zawodny ym. 2004, 49.)

Itse testausohjelmistossa on omat hyvät ja huonot puolensa:

Hyvät puolet:

- Varmasti kaikkien saatavilla (osa lähdekoodipakettia)
- Ollut käytössä jo kauan, vertailukelpoista materiaalia

Huonot puolet:

- Yksisäikeinen
- Vaatii Perl-ohjelmointikielen ja useita kirjastoja
- Ei välttämättä vastaa kaikissa tilanteissa realistisia tilanteita

Testeissä voi keskittyä johonkin yksittäiseen osaan testipaketista tai sitten on mahdollisuus ajaa kaikki testit peräkkäin. Omassa projektissamme ajoimme kaikki testit, jolloin saimme kaikkien yksittäisten testien tulokset ja voimme verrata niiden tuloksia toisiinsa. Pidimme testeissä käytetyt parametrit aina samoina, jolloin voimme olla varmoja tulosten vertailukelpoisuudesta. Saimme samalla testeissä kuluneen kokonaisajan, joka on tärkeä tieto asetettaessa palvelimia paremmuusjärjestykseen.

4.3.1 ATIS-testi

ATIS-testissä luodaan aluksi tietokannan taulut ja samalla syötetään niihin testeissä käytettävä informaatio. Tietokantana toimii kuvitteellinen esimerkki, jossa simuloidaan lentokentällä tapahtuvia tietoteknisiä toimintoja. Simuloinnissa on mukana mm. lentoyhtiöitä, lentokonemalleja, lentokenttiä ja kaupunkeja. Näiden simuloitujen kyselyjen avulla saadaan tietokantapalvelinta rasitettua simuloimalla lähes oikeita tietokantatoimintoja.

4.3.2 Alter table-testi

Alter table-testissä käytetään tietokantojen rakenteellisen muutoksen tekevää Alter table-käskyä (van der Lans 1988, 35). Ajoissa muutetaan tietokannan taulujen rakennetta niin, että oletusarvoisesti yhteensä tuhat tietokannan riviä muutetaan sata kertaa. Tässä testissä tulee hyvin esille tietokannan levynkäsittelyn nopeus, koska tiedot joudutaan päivittämään todella useasti. Tässä testissä kärsivät erityisesti virtualisoidut koneet, joiden levynkäsittelynopeus ei pärjännyt perinteisille palvelimille.

4.3.3 Big tables-testi

Testi on melko yksinkertainen rasitusajo, jossa tietokantaa rasitetaan Select- ja Insert SQL-auseilla. Itse tietokannan taulut ovat isoja, josta myös nimi big tables (isot taulut) nimi juontaa. Isojen tietokantataulujen käsittelyssä korostuu palvelimen sisäinen muistin käsittely ja siihen käytetyn ajan, eli nopeuden mittaaminen. Palvelimissa, joissa on nopeat muistiväylät ja tarpeeksi saatavilla olevaa muistia, menestyivät tässä testissä erinomaisesti.

4.3.4 Connect-testi

Connect-testeissä luodaan tietokannan asiakasohjelmalla yhteys itse tietokantaan, lähetetään tietokannalle määrämittaista dataa käsiteltäväksi ja mitataan toimitukseen kulunut aika. Jos asiakasohjelma ja palvelin sijaitsevat samassa fyysisessä verkossa, kuten meidän tapauksessa, on tämän testin tulokset melko merkityksettömiä. Yleensä ongelmat asiakasohjelman ja palvelimen välillä johtuvat palomuureista tai muista fyysisen verkon mukana tulevista ongelmista. Näissä testeissä pärjäsivät fyysiset palvelimet paremmin, johtuen suorasta tuesta verkkolaitteiden välillä.

4.3.5 Create-testi

Testi luo tietyn määrän tauluja, joita meidän testeissämme tehtiin 10000, laskee niiden määrän (vaikka määrä tiedetäänkin) ja lopuksi poistaa DROP-komennolla taulut tietokannasta. Create-testissä saadaan esille koneiden levykäsittelyyn kuluva aika, joka on varsinkin luku- ja kirjoitustilanteissa äärimmäisen tärkeä ominaisuus. Kuten muissakin testeissä, mitä pienempi aika testeihin kuluu, niin sitä paremmat pisteet palvelin kerää kokonaistesteissä.

Myös tässä testissä kuvastui hyvin, miten paljon hitaampia käytössämme olleet virtualisoidut palvelimet ovat. Jopa meidän testeissä ollut dedikoitu virtuaalikone hävisi tässä testissä mielestämme jopa niin paljon, että toivoimme laitteistoylläpidolta kommentteja asiaan.

4.3.6 Insert-testi

Insert-lause lisää yhden tai useamman rivin tietokantaan. Yksinkertaisesti voidaan todeta, Insert-lausella siis kirjoitetaan tietoa tietokantaan. Tässä testissä tietokantaan luodaan yksinkertainen esimerkkitaulu, johon talletetaan 100000 tietuetta tietyssä järjestyksessä. Tämän jälkeen samaan kantaan talletetaan samat 100000 tietuetta päinvastaisessa kuin edellisessä, jolloin saadaan varmuus, ettei tietueita ole jäänyt välimuistiin sotkemaan kirjoitusta välimuistin puskurista levylle. Viimeiseksi nuo samat tietueet talletetaan vielä kerran sattumanvaraisessa järjestyksessä tietokantaan. Insert-testeissä selviää lähinnä, miten nopeasti palvelin suoriutuu tietojen kirjoittamisesta levylle. Nopeat kovalevyt ovat tässä testissä selvässä suosikin asemassa ja se näkyi myös lopputuloksissa. Xeon- ja Opteron-prosessoreilla varustetut Blade-palvelimet suoriutuivat molemmat erinomaisesti testeistä, kun taas virtualisoidut palvelimet kärsivät selkeästi niiden sisäisesti virtualisoiduista levytoiminnoista.

4.3.7 Select-testi

Tärkein ja useimmiten käytetty SQL-kyselykielen komento on Select. Select-komentoa käytetään tiedon hakemiseen tietokannan tauluista ja Select-komennon tuloksena saadaan aina määrämittainen taulu. Tällaista tulosta käytetään esimerkiksi raporteissa. (van der Lans 1988, 45.)

Select-testimodulissa rasitetaan tietokantaan luotuja testitauluja useilla tuhansilla hauilla, jotka ovat rakenteeltaan hyvin monimutkaisia. Monimutkaisten Select-lauseiden takia tietokantaan kohdistuu hyvin suuret rasitukset ja yleensä myös hakutulosten ulossaaminen kestää normaalia kauemmin. Parhaiten Select-testeissä menestyi Opteron-prosessorilla varustettu palvelin, joka läpäisi kaikki testit 171 sekunnissa (taulukko 3). Virtualisoidut koneet pärjäsivät yleisesti ottaen heikosti tässä testissä.

4.3.8 Wisconsin-testi

Wisconsin-testit on alun perin kehitetty PostgreSQL-tietokannalle kehitetty testimoduli, jolla siis haluttiin testata PostgreSQL:n tietokantaa. Testissä toistetaan jo aiemmin tehtyjä testejä, joissa aluksi luodaan tietokanta, täytetään se testidatalla ja tämän jälkeen tietokantaan kohdistetaan Select-hakuja. Kokonaisvaltainen kuormitus saadaan aikaan, kun kaikki testin osiot toistetaan useaan otteeseen. Testeissämme yksittäisiä testikertoja oli kymmenen ja ne toistettiin Wisconsin-testeissä peräkkäin. Tuloksissa erot saatiin suuriksi, fyysisten palvelinten hyväksi. Erot olivat parhaimman ja heikoimman palvelimen välillä melkein kymmenkertaiset.

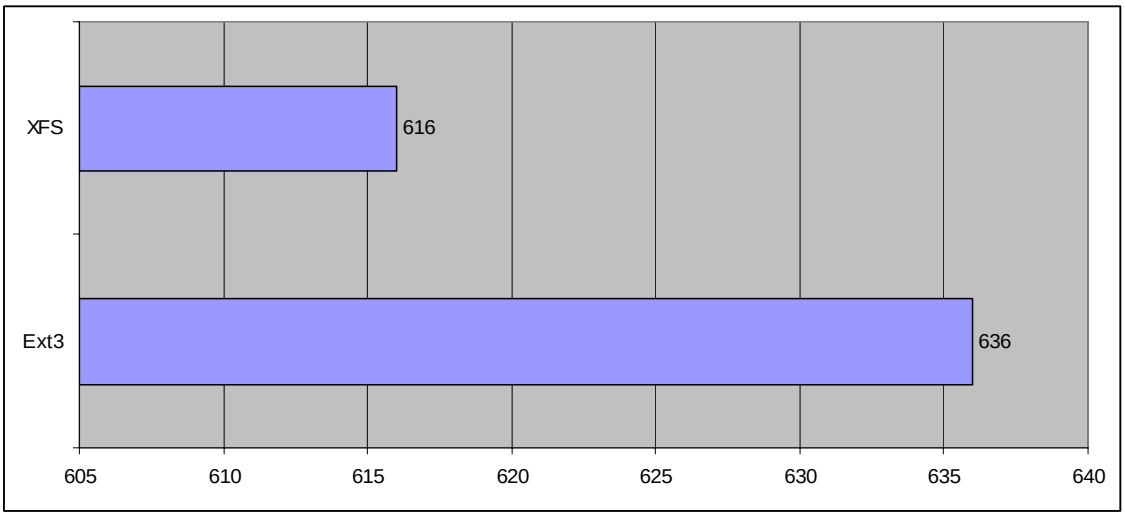
4.4 Tiedostojärjestelmän testaus

Red Hat Enterprise Linux-käyttöjärjestelmän oletustiedostojärjestelmänä on Ext3-niminen tiedostojärjestelmä. Se on oletuksena myös monissa muissa Linux-käyttöjärjestelmäjakeluissa johtuen hyvästä suorituskyvystä ja eritoten vikasietoisuudestaan. (MPoli Oy Ext3 2008.)

Saimme palvelintestimme valmiiksi sen verran nopeasti, että meille jäi projektin puitteissa aikaa myös testata tiedostojärjestelmiä. Olimme halukkaita kokeilemaan muitakin tiedostojärjestelmiä kuin Ext3, koska sen suorituskyky ei välttämättä ole tehokkain. Red Hat Enterprise Linuxin tukee useita eri tiedostojärjestelmiä, jotka voisivat sopia käyttöömme mutta päätimme testata XFS-tiedostojärjestelmää, jonka hyvästä toiminnasta olimme saaneet vinkkejä kollegoiltamme.

XFS-tiedostojärjestelmä on erittäin suorituskykyinen kirjaava (Journaling) tiedostojärjestelmä, jonka alun perin on kehittänyt Silicon Graphics heidän IRIX-käyttöjärjestelmälleen. Myöhemmin XFS on muokattu myös Linux-käyttöjärjestelmään sopivaksi. XFS:n paras puoli on sen kyky hallita todella suuria tiedostoja ja niiden siirtoa levyalueilla. (MPoli Oy XFS 2008.)

Tiedostojärjestelmänä XFS osoittautui hieman hankalaksi asentaa CSC:n tuettuun Linux-ympäristöön ja sen suorituskykykään ei vastannut odotuksia testeissämme. Toki pieniä eroja saatiin aikaan, mutta erot olivat hyvin marginaaliset (taulukko 1). Tulosten perusteella päätimme asentaa uuteen palvelimeemme Ext3-tiedostojärjestelmän. Testien tuomat erot olisivat voineet olla isompia, jos käytettävissä oleva testiaineisto olisi ollut laajempaa. Silloin olisi XFS:n parhaimpana ominaisuutena mainittu suurten tiedostojen käsittely tullut paremmin esille. Tämä lähestymistapa olisi kuitenkin sotkenut testien objektiivisuutta, joten emme lähteneet muokkaamaan omia testejämme saadaksemme jonkin tietyn tiedostojärjestelmän näyttämään paremmalta kuin se oikeasti on. Ext3 on kuitenkin hyvä tiedostojärjestelmä ja on niin monikäyttöinen, että voimme olla varmoja, että se tulee toimimaan laitteistossamme hyvin.

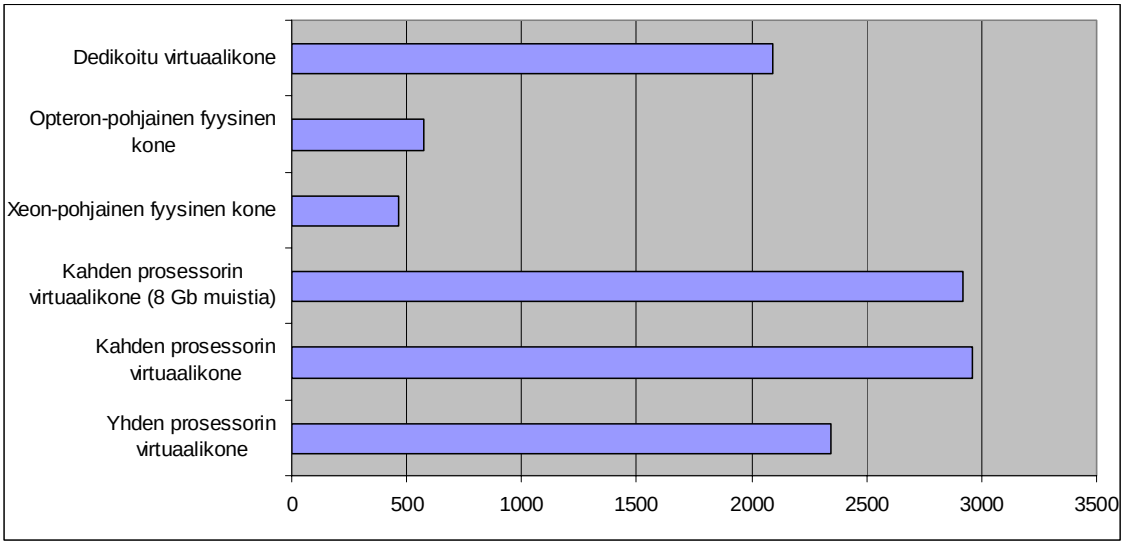


Kuva 2: Tiedostojärjestelmien testiin kuulunut aika sekunteina mitattuna

4.5 Laitteistojen testitulokset

Laitteistojen testit vahvistivat näkemyksiämme eri vaihtoehtojen suorituskyvyistä. Olimme aiemmissa projekteissa todenneet virtualisoitujen koneiden käyttökelpoisuuden useissa eri tilanteissa, kuten tulostuspalvelimina tai palomuuripalvelimena. Jos taas lähtökohtana on todella tehokkaan koneen valjastaminen tietokantakäyttöön, ei virtualisoitu ympäristö ole oikea valinta.

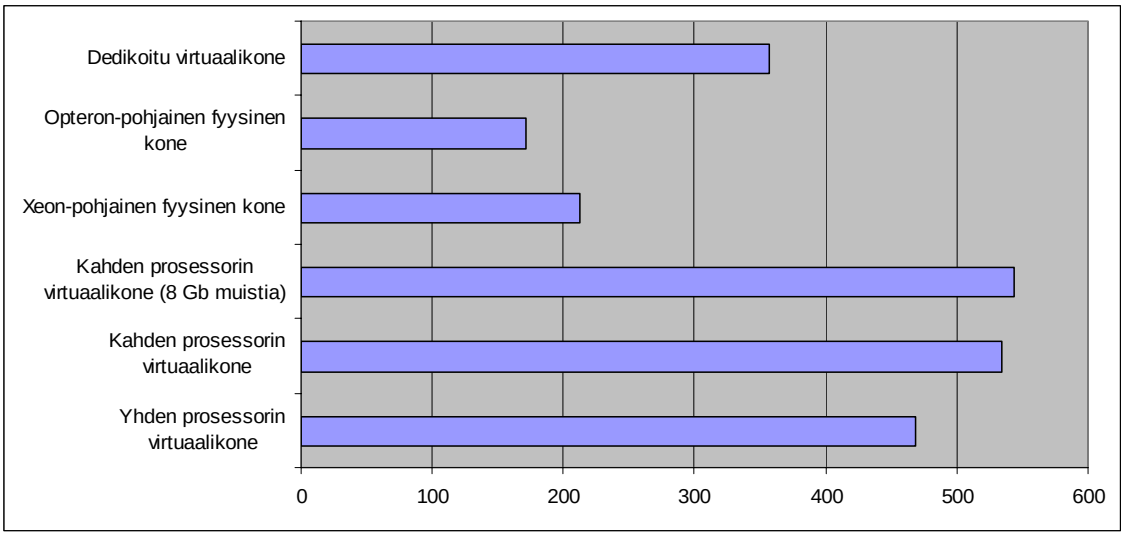
Kuten Insert-testitulosten (taulukko 2) testitulosten perusteella voidaan havaita, on virtualisoitu palvelinympäristö huomattavasti hitaampi kuin fyysinen palvelinlaitteisto. Suurin osa käytetystä ajasta menee levytoimintojen tekemiseen ja kun käytössä ei ole suoraa kovalevyyhteyttä, niin tämä aiheuttaa suorituskykyongelmia. Tämä tuli hyvin esille kahdeksan gigatavun muistilla varustetun koneen testeissä. Vaikka muistimäärä oli sama kuin fyysisten laitteiden, niin suorituskyky ei siltikään ollut hyvä. Muistia olisi luultavasti voinut lisätä jopa 16 gigatavuun asti, eikä suorituskyvyssä olisi ollut merkittäviä eroja.



Kuva 3: Insert-testitulokset. Aikajana kuvaa testeihin kulunutta aikaa.

Suurin osa tietokantoihin kohdistuvista tapahtumista on tietojen hakua itse kannasta. Nämä kyselyt tehdään useimmiten SQL-kielen Select-lauseella, jota käytimme testeissämme (taulukko 3). Testeissä kävi ilmi, että Select-lauseissa oli virtualisoitu palvelin heikoimmin menestyvä palvelinalusta. Tuloksissa voi hieman hämmentää se, että yhden prosessorin kone suoriutui ajetuista testeistä nopeammin kuin kahden prosessorin kone. Tämä johtuu siitä, että testeissä käytössä oleva ohjelmisto osaa käyttää vain yhtä prosessoria ajaessaan testejä läpi.

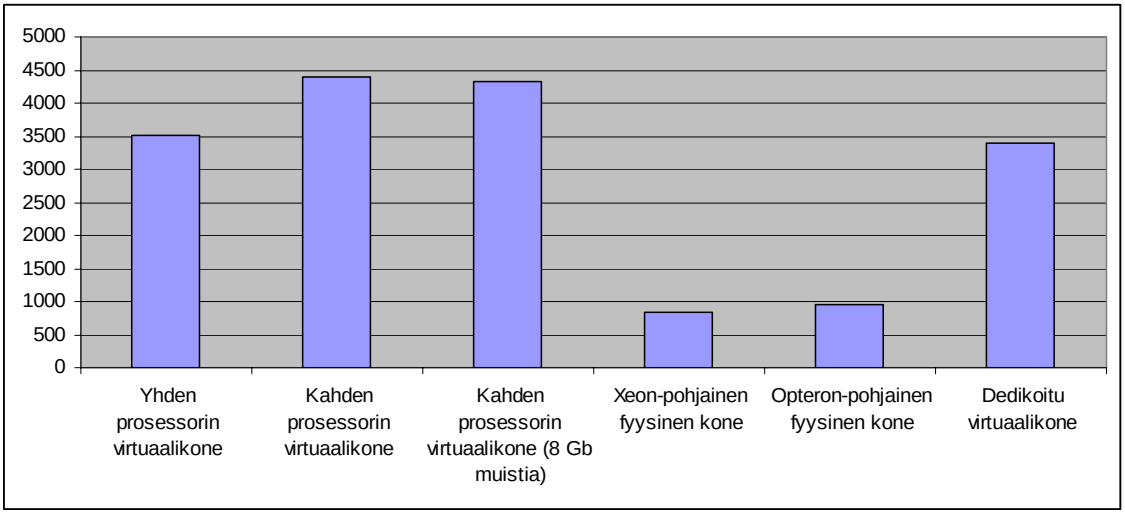
Fyysiset palvelimet suoriutuivat Select-lauseisiin perustuvista testeistä kolme-neljä kertaa paremmin, joten voittajan valitseminen tässä testiosuudessa ei ollut kovin vaikeaa. Eroa ei voi pistää täysin VMware-virtualisointiohjelmiston syyksi, vaan siihen vaikuttivat suuresti käytössä olleet alustapalvelimet. Täysin yhdelle koneelle dedikoitu virtuaalikone hävisi tehoissa noin kaksinkertaisesti verrattuna fyysisiin palvelimiin.



Kuva 4: Select-kyselyjen testitulokset. Aikajana kuvaa testeihin kulunutta aikaa.

4.6 Testitulosten päätelmät ja hankintaehdotus

Kokonaistulosten perusteella voittajaksi selviytyi ylivoimaisesti Xeon-prosessorilla varustettu fyysinen Hewlett-Packardin palvelin (taulukko 4). Eroa Opteron-prosessorilla toimivaan lähes samankaltaiseen malliin saatiin eroa noin 10 %. VMware-tekniikalla virtualisoidut palvelimet osoittautuivat suorituskykytesteissä niin heikoiksi vaihtoehdoiksi, ettei niitä voi harkita hankittavaksi.



Kuva 5: Testien kokonaistulokset. Aikajana kuvaa testeihin kulutta kokonaisaikaa.

Testien osoittaessa Xeon-prosessorilla varustetun kokoonpanon olevan suorituskyvyllisesti testien voittaja, oli perusteltua projektiryhmän ehdottaa hankittavaksi juuri tätä kokoonpanoa. Laitteistojen hankinnoista vastaavan Pata-ryhmän kanssa keskusteltaessa kävi myös ilmi, että

haluttu Xeon-valinta olisi heille sopiva vaihtoehto. CSC:lle oli juuri hankittu palvelimia, joissa oli testeissä ollut Xeon-prosessori ja kuulimme, että nämä laitteistot ovat toimineet käytössä erinomaisesti. Niinpä projektiryhmä päätti ehdottaa, että hankkisimme, hyvien käyttökokemusten tullessa ilmi, Xeon-prosessoreilla varustetun palvelimen.

Hankittavaksi ehdotettu palvelin sisältää seuraavat komponentit:

- HP ProLiant DL380 G5 - 2.66GHz Rack Server - 1P
- Prosessorina Quad-Core Intel Xeon E5430 (2.66GHz, 1333MHz FSB, 80W)
- Muistia 16 Gigatavua
- Levyohjain HP Smart Array P400/256
- Verkkokorttina sisäänrakennettu NC373i Multifunction Gigabit Network Adapter
- ja 412648-B21 HP NC360T PCI Express Dual Port Gigabit Server Adapter - Low Profile
- Kuitukorttina A8003A HP Fibre Channel 2242SR 4GB PCI-e HBA – Low Profile
- Virtalähteenä HP 1000-W Hot-Plug Power Supply
- Paikallisina kovalevyinä HP 146 GB Hot Plug 2.5 SAS Dual Port 10,000 rpm Hard Drive
- Takuu 4 vuotta, korjaus paikan päällä ja vasteaika 4 tuntia ja takuu voimassa 24x7

5 PALVELIMEN ASENNUS

Kun valitsemamme ja tilaamamme palvelin on viimein saatu fyysisesti toimitiloihimme, voi seuraava vaihe, eli palvelimen asentaminen viimein alkaa. Pienpalvelinten asennus on CSC:llä keskitetty Palvelutuotantoalustat ryhmälle, joka hoitaa asennuksen alusta alkaen. Ryhmä hoitaa itse laitteiden tilauksen, jolloin heillä on suuri vastuu meidän projektissamme. Koska palvelu on keskitetty, niin pyrkimyksenä on että kokoonpanot vakioidaan mahdollisimman pitkälle, ettei konekanta kasva organisaatiossamme liian kirjavaksi. Myös mahdolliset vikojen korjaamiset ja takuujärjestelyt hoituvat heidän kauttansa. Vakiointi helpottaa ylläpidon tehtäviä ja takaa jatkuvuuden laitteistojen toiminnassa. Eksoottiset palvelinkokoonpanot ovat aiemmin aiheuttaneet suurta päänvaivaa ja tällaisista toimintamalleista on onneksemme päästy eroon.

Palvelutuotantoalustat-ryhmä on myös suorassa yhteydessä muihin tarvittaviin CSC:n sidosryhmiin ja hoitaa tarvittavat tukipyynnöt sidosryhmille. Sidosryhmistä aivan ensimmäiseksi mukaan tulee konesaliryhmä, joka vastaa CSC:llä olevasta tietokonesalista, jossa suurin osa palvelimistamme sijaitsee. Heidän tehtävänään on määritellä uuden palvelimen fyysinen paikka, johon vaikuttaa esimerkiksi katkeamattoman sähkönsaannin mahdollisuus, palvelimen jäähdytys ja tietoliikennekaapeloinnin saatavuus. Datan tallennuspalveluryhmä hoitaa tietokantamme käyttämästä huippunopeasta verkkolevystämme ja tietoliikenneyhteyksien määrittelyistä, muokkauksista ja asennuksista vastaa CSC:n Funet-palvelujen ryhmä.

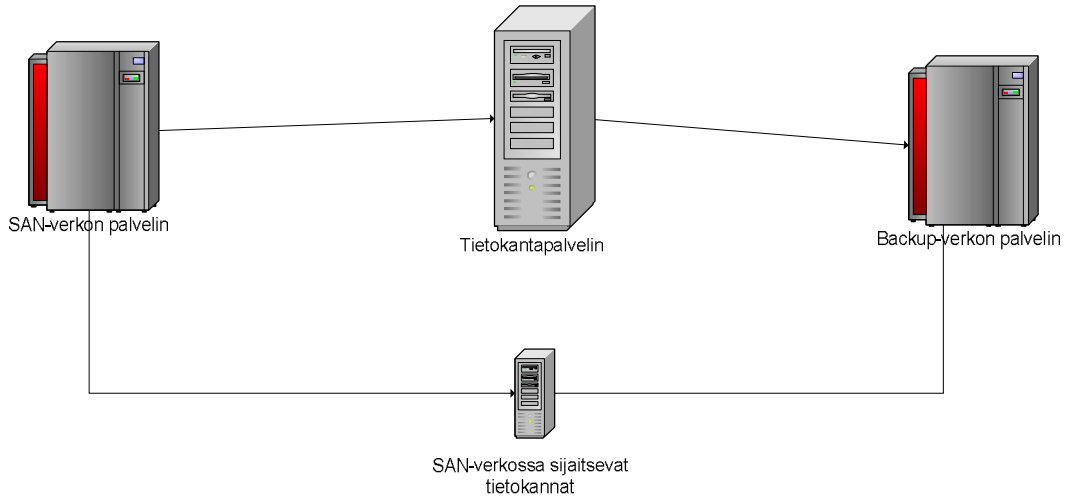
5.1 Käyttöjärjestelmän asennus

Käyttöjärjestelmän asennus on automatisoitu hyvin pitkälle Linux-käyttöjärjestelmien kehittämiseen erikoistuneen Red Hat-yhtiön toimesta. CSC:lle on hankittu heidän RHN Satellite Server tuote, jonka avulla voidaan suorittaa esimerkiksi käyttöjärjestelmien asennusta, päivitys ja tarkkailu. ”RHN Satellite Server tarjoaa organisaatioille ratkaisun, joka tarjoaa täydellisen kontrollin ja tietoturvallisen vaihtoehdon ohjelmistojen asentamiseen palvelimiin. Se tarjoaa Red Hat Network-asiakkaille ketterimmän mahdollisen tuotteen, jolla palvelimet voidaan pitää tietoturvallisina ja päivitettyinä.” (Redhat Inc 2008.)

Uusin, luotettavaksi todettu käyttöjärjestelmä RHN Satelliteen oli asennushetkellä Red Hat Enterprise Linux Server release 5.3, koodinimeltään Tikanga. Asennus tehtiin etäasennuksena, jolloin kone käynnistettiin Cd-rom-levyn avulla ja itse asennuspaketit noudettiin asennuspalvelimelta verkon yli. Asennus sinänsä tapahtuu ongelmattomissa tapauksissa todella nopeasti. Käyttämämme palvelin ei sisältänyt eksoottisia komponentteja, joten meidän asennuksemme oli hyvin suoraviivaista ja nopeaa.

5.2 Palvelimen asennus varmuuskopiointi- ja SAN-verkkoon

Palvelimemme pitää asentaa CSC:n sisäiseen SAN-verkkoon (Storage Area Network), jossa käyttämämme levypalvelimet sijaitsevat. Levypalvelimella sijaitsevat kehittämämme palvelumme tietokannat, koska niiden säilyttäminen paikallisella levyllä ei olisi kustannussyistä järkevää. SAN-verkko on täysin muusta tietokoneverkosta erillinen tallennusverkko, jossa liikkuu vain levyiltä koneille menevä ja takaisin menevä data. Tietojen saatavuus paranee, koska ne ovat useiden palvelinkoneiden saatavilla samalla kertaa.



Kuva 6: Tietokantapalvelimen, SAN-verkon palvelimen ja Backup-verkon palvelimen yhteistoimintaa havainnollistava malli

MySQL-tietokantojen varmuuskopiointi on todella tärkeää luotettavan palvelun ylläpitämiseksi. Tällä hetkellä MySQL-tietokannoille ei ole saatavilla järin montaa varmuuskopiointityökalua, jotka osaavat hoitaa kopioinnin hyvin ja luotettavasti. Miksi sitten kannattaa tehdä tämä useimmille tietokantojen ylläpitäjille hyvin epämieluisa tehtävä? Suurin motivaatio pitäisi olla toimivan järjestelmän sujuva käyttö, mutta mielessä kannattaa pitää, että mikään järjestelmä ei ole täysin luotettava ja ongelmia voi ilmaantua monissa eri komponenteissa. Minkälaisiin ongelmiin kannattaa sitten varautua? Ongelmia aiheuttavat tapahtumat, joissa merkittävät määrät tietoa korruptoituvat, katoavat tai ovat muuten käyttökelvottomia. Tällaisia tapauksia voivat aiheuttaa esimerkiksi:

- Laitteiston vaurioituminen
- Ohjelmistojen vaurioituminen
- Tahaton tietojen poisto
- Fyysisesti tuhoutunut palvelinlaitteisto (esim. tulipalo tai vesivahinko)

(Zawodny ym. 2004, 187-188)

Palvelimien varmuuskopiointi on CSC:n organisaatiorakenteessa annettu Datan tallennuspalvelut-ryhmän tehtäväksi. Käytössä on täysin automaattinen, keskitetty ja luotettava varmuuskopiointijärjestelmä, joka on eritytetty omaan tietoturvalliseen verkkoonsa (kuva 2). Verkkoon ei voi ottaa ulkopuolisia yhteyksiä, joten voimme olla melko varmoja liikenteen häiriöttömästä ja ennen kaikkea turvallisesta kulusta. Varmuuskopiointia varten asensimme asiakasohjelmiston, joka ottaa etäkomentonsa vastaan varmuuskopioinnin suorittavalta palvelimelta. Kuormituksen takia on talomme eri pienpalvelimet ajoitettu toimimaan ympärivuorokautisessa valmiustilassa ja sen takia varmuuskopioinnin ajoitus on hoidettu täysin palvelimen puolesta. Pääsääntöisesti palvelinten tiedot varmuuskopioidaan kerran vuorokaudessa. Tällöin suoritetaan ns. incremental-kopiointi, eli vain muuttuneet tai täysin uudet hakemistot ja tiedostot talletetaan. Palvelinten täydellinen varmuuskopiointi suoritetaan mutta tästä sovitaan yleensä palvelun omistajan kanssa. Uuden palvelumme täydellistä varmuuskopiointia ei kovin usein suoriteta, koska itse palvelimen tiedot eivät muuta kovinkaan usein. Tietokantojen varmuuskopiot tehdään päivittäin, koska niiden tiedot voivat muuttua useastikin päivän aikana.

5.3 MySQL-asennus

MySQL on open source-pohjainen tietokanta, mikä tarkoittaa sitä, että sen lähdekoodi on kaikkien saatavilla ja muokattavissa täysin ilmaiseksi. Käyttöjärjestelmät kuten Linux ja FreeBSD ovat open source-lisenssin alaisia tuotteita. Windows-käyttöjärjestelmä on taas täysin kaupallinen tuote, jonka lähdekoodin omistaa, kehittää ja hallinnoi Microsoft. MySQL:n lähdekoodi on vapaasti saatavilla ja sen asennuksessa tarjolla kaksi eri vaihtoehtoa:

- Asentamalla binääri-version, mikä tarkoittaa MySQL:n kehittäjät (tai joku kolmas osapuoli) ovat valmiiksi kääntäneet lähdekoodin halutulle käyttöjärjestelmäalustalle.
- Asentaminen lähdekoodista kääntämällä, jolloin MySQL-lähdekoodi käännetään ja asennetaan itse halutulle alustalle

Binääri-version asentaminen on yleistäen ehdottomasti helpompi ja nopeampi tapa asentaa MySQL, mutta asennustavan valintaa vaikuttaa monet asiat, kuten asentajan tietämys ja osaaminen. Aloittelevalle järjestelmäylläpitäjälle ehdottomasti helpompi vaihtoehto on binääri-version asentaminen, koska siinä mahdollisten virheiden aikaansaaminen on minimissään. Jos kuitenkin haluaa vaikuttaa oman tietokantaympäristönsä sisältöön, on lähdekoodista kääntäminen ja asentaminen paras vaihtoehto. Tällöin voi vaikuttaa lähes kaikkiin tietokannan vaihtoehtoihin ja samalla oppii ymmärtämään enemmän tietokannan sisäisiä toimintoja. (Gilfillan 2003, 451-452.)

Projektissamme päädyimme asentamaan MySQL-tietokannan lähdekoodista, koska se on ollut CSC:llä vakiintunut toimintapa ja halusimme jatkaa tätä hyvin toimivaa tapaa. Lähdekoodista

asentaminen on työnsä osaavalle järjestelmäasiantuntijalle sangen helppo toimenpide mutta uusien versioiden ilmestyessä, on oltava kuitenkin tarkkaavainen. Muutoksia esimerkiksi käännösparametreihin tulee silloin tällöin ja niihin kannattaa tutustua huolellisesti. Oletusarvoisesti uudet parametrit ovat turvallisia, mutta tietoturvan kannalta ei voi koskaan olla liian varovainen. Kääntämisessä tarvitaan seuraavat komponentit:

- Unix-pohjainen käyttöjärjestelmä (meillä RedHat Enterprise Linux)
- Kopio lähdekoodista, jonka voi hakea verkosta
- Gunzip- ja tar-pakkausohjelmat, jolla lähdekoodipaketti saadaan auki
- Tarpeeksi levytilaa, että asennus onnistuu
- Unix-ympäristön make-komento ja C- tai C++-kääntäjä

Lähdekoodista asentaminen on ohjeita lukemalla ja niitä seuraamalla hyvin suoraviivaista toimintaa. Haluttu asennuspaketti haetaan lähimmältä MySQL-tietokantaa tarjoavalta toimittajalta ja se puretaan haluttuun hakemistoon palvelimella. Asennuspaketti puretaan, tarvittaessa muokataan määrittelytiedoston parametreja ja tämän jälkeen lähdekoodi käännetään palvelimella. Käännöksen onnistuessa voidaan itse tietokanta asentaa haluttuun polkuun palvelimella ja käynnistää se.

Ongelmia asennuksessa tapahtuu valitettavan usein. Periaatteessa asentamisen ei pitäisi olla vaikeaa, mutta tietotekniikassa on hyvin paljon erilaisia kompastuskiviä joihin törmää juuri silloin kuin niihin ei halua törmätä. Kokeneet järjestelmäylläpitäjät osaavat useimmiten kohdistaa ja ratkaista ongelmat helpostikin mutta aloittelijoille voi olla edessä monta harmaita hiuksia tuottavaa tuntia. Useimmista ongelmista löytyy usein kuvaukset MySQL-tietokannan keskustelupalstoilta ja niistä löytyy hyvin usein myös ratkaisut ongelmatilanteisiin. Näitä keskustelupalstoja kannattaa pitää silmällä, jos on usein tekemisissä MySQL:n kanssa. (Gilfillan 2003, 455.)

6 MYSQL-YMPÄRISTÖN MUKAUTUS

MySQL-ympäristön mukautus on tehtävä lähestulkoon aina, koska jokainen palvelinympäristö on erilainen ja tarvitsee aina hieman erilaisia säätöjä ja muokkauksia. Suoraan paketista asennetut ohjelmistot ovat yleensä varsin käyttökelvottomia ja vaativat useasti monen tunnin työn, jolloin palvelin ja ohjelmistot saadaan toimimaan halutulla tavalla. Mukautus meidän tapauksessamme on CSC:n yhteisten komponenttien asentamista ja niiden testaamista. Ympäristössämme on muutamia perustyökaluja, jotka vaativat tarpeellisen muokkauksen, että integrointi ympäristöömme sujuu ongelmitta.

Monet käyttämämme CSC:n sisäiset komponentit asentuivat alustallemme täysin ilman ongelmia. Esimerkiksi MySQL:n toiminnanvarmistukseen käytetty valvontaohjelmisto saatiin asennettua täysin oletusasetuksilla, jotka oli meille ohjeistettu. Toiminnanvarmistuksessa tarkkaillaan tietokannan saatavuutta, käyttöastetta ja useita muita informatiivisia lisätietoja, joita saadaan tarkkailuohjelmistostamme. Ohjelmistolla saamme lähes reaaliaikaista tietoa tietokantamme tilasta ja mahdolliset ongelmatapaukset tulevat tietoomme välittömästi erillisen selainliittymän kautta tai suoraan sähköpostilla.

Tietokantamme muutokset ovat hyvin pieniä ja yksityiskohtaisia. Dbadm-prosessissa seuraamme omaa asennusohjettamme, jossa käydään läpi tietokannan mukautuksen perusasiat. Tähän asennusohjeeseen tehdään muutoksia yleensä aina uusien MySQL:n ominaisuuksien myötä. Muutoksia tietokantaohjelmistoon tulee vaihtelevalla nopeudella ja niistä tiedotetaan useimmiten hyvin. On kuitenkin hyvin paljon oman harrastuneisuuden varassa, että pysyy kaikkien uusien muutoksien mukana. Erityisesti tämä koskee kolmannen osapuolien ohjelmistojen kanssa, jotka vaikuttavat tietokannan toimintaan. Hyvin tarkkaavaisena pitää olla tietokannan tietoturvallisuuteen vaikuttavien ominaisuuksien työstämisessä. Uusien ominaisuuksien välitön käyttöönotto ei ehkä ole suositeltavin askel, koska niiden luotettavuudesta ei välttämättä ole täysin oikeaa tietoa saatavilla. Usein tällaisten uusien ominaisuuksien käyttöönotto tuotannossa on järkevää vasta muutaman ohjelmistopäivityksen jälkeen, jolloin mahdolliset ongelmakohdat on löydetty ja korjattu.

6.1 MySQL-ympäristön turvallisuus

MySQL-tietokannan turvallisuuden ylläpitäminen on äärimmäisen tärkeää, varsinkin kun tiedossa on, että tulemme tallettamaan omien asiakkaidemme tärkeitä tutkimustuloksia. Tällöin olemme myös vastuussa heidän tiedoistaan, joten tietoturvallisen ympäristön luominen on yksi perusedellytys, ennen kuin palvelu avataan asiakaskäyttöön.

Meidän tapauksessamme ensimmäinen prioriteetti oli turvallisen ympäristön luominen tietokantapalvelullemme. Aloitimme tämän työvaiheen palomuurien säätämisellä. Oletusarvoisesti MySQL ei ole suojattu ja sen avoin TCP-portti (3306) on täysin avoinna ulkopuoliselle maailmalle, eli meidän tapauksessa Internetiin. Palvelimemme palomuuri muokattiin MySQL:n osalta niin, että siihen pystyy ottamaan yhteyksiä vain CSC:n omasta laskentaympäristön verkosta. Tällä tavoin voimme varmistaa, ettei palveluumme tule ainakaan talon ulkopuolisia murtoyrityksiä palvelumme kautta. Tarkastimme samalla, että palvelimemme kaikki muut portit ovat kiinni, eikä niihin ole sallittu CSC:n ulkopuolisia yhteyksiä.

Käytössämme olevan käyttöjärjestelmän tietoturvallisuudesta emme joudu ottamaan pääasiallista vastuuta, koska sen hoitaa CSC:n organisaation palvelutuotantoalustat-ryhmä. Tosin olemme linjanneet, että tulemme myös itse huolehtimaan käyttöjärjestelmän turvallisuudesta, koska emme voi olla täysin riippuvaisia muiden valppaudesta. Tämän lisäksi olemme asentaneet käyttöjärjestelmän päälle useita muita ohjelmistoja, joiden turvallisuudesta vastaamme itse.

6.2 Käyttäjähallinnon kehittäminen

Uuden järjestelmän integroiminen jo olemassa oleviin ympäristöihimme oli haastava tehtävä. Emme halunneet keksiä pyörää uudelleen ja tehdä esimerkiksi käyttäjähallintaa uudelleen vaan halusimme käyttää jo olemassa olevia CSC:n tarjoamia komponentteja. Komponenttien yhdistäminen ei ole täysin automaattinen toiminto, koska MySQL:n sisäinen käyttäjähallinta on sangen puutteellinen. Jouduimme tekemään useita eri muokkauksia joihinkin sisäisiin rakenteisiin ja nämä muutokset jouduttiin tekemään MySQL:n päässä, koska emme halunneet muokata CSC:n omia komponentteja. CSC:n omien komponenttien muokkaus olisi voinut aiheuttaa yhteensopivuusongelmia muissa järjestelmissä, joten oli turvallisinta tehdä tarvittavat muutokset uuden järjestelmän piirissä.

CSC:llä käytössä oleva käyttäjähallinta on pääosin kehitetty itse ja se on käytössä erityisesti Unix-palvelinten puolella. Sen suurin hyöty on sen ketteryys ja muokattavuus omien tarpeidemme mukaisesti. Olimme alustavasti päättäneet, että käytämme tätä samaa, jo valmista ja toimivaa käyttäjähallintaa projektissamme, mutta törmäsimme testivaiheessa ongelmiin. Käyttäjien tunnistetietojen ja salasanojen siirto MySQL:ään ei toiminut halutusti ja emme saaneet sitä toimimaan kunnolla. MySQL käyttää täysin omaa menetelmää käyttäjien tunnistamiseen ja hallintaan, jolloin jouduimme luopumaan suunnitelmistamme.

Koska Unix-palvelinten käytössä oleva käyttäjähallinta osoittautui meille käyttökelvottomaksi, päätimme käyttää MySQL:n omaa sisäistä käyttäjähallintaa. Tosin joudumme tekemään siihen

joitain muutoksia, että saamme projektillemme asetettuja vaatimuksia toteutetuksi. Käyttäjien luonnin päätimme toteuttaa olemassa olevia sisäisiä suosituksia mukaillen, eli Perl-ohjelmointikielellä toteutetuilla lyhyillä ohjelmanpätkillä. Koska käyttäjähallinto on Usermgr-prosessin hoidossa, halusimme tehdä heidän työstään mahdollisimman helppoa. Tämä tarkoittaa käytännössä sitä, että myös meidän projektimme käyttäjähallintoa voidaan hoitaa samalta koneelta, jolla hallinnoidaan CSC:n muitakin käyttäjähallinnon prosesseja. Prosessissa annetaan yksi komento, jolla luodaan yhdelle projektille kolme eritasoista käyttäjätunnusta. Eritasoisissa tunnuksissa on se hyvä puoli, että niiden käyttäjäoikeuksia voidaan rajoittaa tarpeen mukaan (kuva 3). Meidän mallissamme heikoimmat oikeudet saa read-tunnus, jolla on vain lukuoikeudet tietokantaamme. Toisella tasolla oikeuksia lisätään, jolloin user-tason tunnus saa jo tietokannan muokkaus-, lisäys- ja poisto-oikeudet. Admin-tason käyttäjätunnus saa täydet oikeudet projektin tietokantaan, eli tunnus voi luoda uusia tauluja, poistaa niitä ja tehdä muita ylemmän tason tietokantaoperaatioita.

	db_read	db_user	db_admin
Select_priv	X	X	X
Insert_priv		X	X
Update_priv		X	X
Delete_Priv		X	X
Create_priv			X
Drop_priv			X
References_priv			X
Index_priv			X
Alter_priv			X
Create_tmp_table_priv			X
Lock_tables_priv			X
Create_view_priv			X
Show_view_priv			X
Create_routine_priv			X
Alter_routine_priv			X
Execute_priv			X

Taulukko 1: Tietokanta-asiakkaiden kolmen käyttäjätunnuksen oikeuksien jaottelu

6.3 Käyttäjien kiintiöt

Toinen suurehko puute MySQL:ssä on kiintiöiden (engl. quota) puuttuminen. Tämä tarkoittaa siis sitä, ettei käyttäjällä ja hänen tietokannoillaan ole minkäänlaisia rajoituksia palvelimen resurssien kanssa. Käyttäjä voi siis käyttää kaiken saatavilla olevan levytilan, joka johtaa loppujen lopuksi siihen, että levytila täyttyy, tietokantamoottori jumiutuu ja pahimmassa tapauksessa koko käyttöjärjestelmä kaatuu. Saatavilla oleva prosessoriteho on kaikkien käyttäjien käytettävissä ja tämä puute voi pahimmassa tapauksessa aiheuttaa palvelimen jumiutumisia tai vähintäänkin tuntuvia hidastumisia, jos käyttäjät pystyvät rajattomasti käyttämään palvelimen resursseja.

Näistä kahdesta ongelmasta on levytilan täyttyminen helpommin ratkaistavissa, joten päätimme projektin sisäisessä ryhmäkokouksessa ratkaista ensin tämän ongelman. Koska emme olleet täysin varmoja tietokantapalvelun suosiosta, emme halunneet vielä ottaa kantaa miten prosessoritehojen jakaminen tulisi ratkaista. Mielestämme on tärkeää ratkaista ensin levytilakiintiöiden puuttumisen aiheuttama ongelma, koska se tekee palvelusta välittömästi käyttökelvottoman, jos käytössä oleva levytila täyttyy. Jos palvelu osoittautuu hyvin suosituksi, on varmasti myös prosessoritehon rajoittaminen käyttäjämäärän mukaan sellainen kohta johon meidän on keksittävä jokin kestävä ratkaisu.

Levytilan täyttymiseen ja sen estämiseen on useita eri lähestymistapoja, joita on myös Internetissä vapaasti saatavilla. Yksi varteenotettavin tapa olisi käyttää Unix-pohjaisten käyttöjärjestelmien jo olemassa olevaa quota-levyresursseja tarkkailevan komennon käyttöönotto. Quota-komento tarkkailee käyttäjätunnusten, eli käytännössä käyttäjien, omistuksessa olevia tiedostoja ja laskee niiden summan. Jos summa ylittää ennaltamääritetyn rajoituksen, seurauksena voi olla esimerkiksi ko. käyttäjätunnuksen kirjoitusoikeuksien poistaminen. Kirjoitusoikeudet palautettaisiin jälleen, kun käyttäjä on poistanut rajoituksen ylittävän määrän tiedostoja tai hakemistoja.

Jouduimme hylkäämään ajatuksen Unix-käyttäjätunnuksiin perustuvasta kiintiöjärjestelmästä, koska käyttäjähallintomme ei käyttäjähallinnon mukautuksesta johtuen perustu Unix-tunnuksiin. Tästä johtuen joudumme tekemään oman sovelluksen ratkaisusta, jossa levytilan tarkkailu perustuu CSC:llä olemassa olevaan käyttäjäprojektiin ja sille ennaltamääriteltyyn levytilaan. Sovelluksessa projektille annettua levytilaa verrataan kymmenen minuutin välein varsinaiseen projektin tietokannan levytilaan. Jos tietokannan levytila ylittyy hetkellisesti, niin valvontaohjelmistomme poistaa projektin käyttäjätunnuksilta mahdollisuuden lisätä ja muokata tietoja kannassa. Ylityksestä ilmoitetaan samalla käyttäjälle sähköpostilla ja toivotaan heidän poistavan tarvittavan määrän tietoja kannastaan. Kun ylitys on saatu korjattua, eli levytila on palautettu jälleen alle ennaltamääritetyn maksimiarvon, niin silloin käyttäjän täyden tietokantaoikeudet jälleen palautetaan ja siitä lähetetään sähköposti, jossa tapahtumasta kerrotaan.

Käyttäjillä on mahdollisuus tarkkailla oman projektinsa tietokannan käyttöä rakentamillamme ohjelmilla. Ohjelma on tietokantaan sisäänrakennettu proseduuri, joka laskee kunkin projektin omistamat tietokantojen yhteenlasketun levytilan yhteen ja esittää sen luettavassa muodossa käyttäjälle. Alun perin olimme ajatelleet pelkästään ajastettujen raporttien lähettämistä käyttäjille, kun heidän kiintiönsä on täyttymässä, mutta luovuimme tästä ajatuksesta. On käyttäjän kannalta tärkeää tietää, miten paljon hän on käyttänyt resurssejaan ja miten paljon niitä on jäljellä. Järjestelmässä on vielä olemassa piirre, jossa käyttäjää informoidaan sähköpostilla, kun hänen resursseistaan on 80 % käytetty. Tällöin käyttäjä voi reagoida haluamallaan tavalla tilanteeseen, eli olemme lisänneet käyttäjän mahdollisuuksia tehdä pitkäjänteisiä suunnitelmia tietokannan käytölleen.

6.4 Käyttäjätunnusten vanheneminen

MySQL:n käyttäjähallinta on todella minimalistinen. Tällä on saavutettu ainakin se, että ympäristön muokkaajilla on täysin rajoittamattomat mahdollisuudet ratkaista käyttäjähallinnon avoimia kysymyksiä. Perusasetuksilla on käyttäjätiedoissa talletettuna vain käyttäjän salasana ja käyttäjätunnus. Muita tietoja ei välttämättä tarvitse tallettaa mutta yleisesti on suositeltavaa, että tietokantaan liittävien asiakasohjelmien käytössä olevat IP-osoitteet tai nimipalvelussa olevat nimet talletetaan. Meille nämä tiedot eivät valitettavasti riitä, joten joudumme tekemään järjestelmään mukautuksia, jotka sopivat meillä käytössä olevaan infrastruktuuriin.

Käyttäjähallinnon näkökulmasta kaikilla käyttäjätunnuksilla on elinkaari, jossa käyttäjätunnus luodaan ja jossain vaiheessa käyttäjätunnus poistetaan järjestelmästä. Meille tämä teetti työtä, koska tietokannan käyttäjähallinnossa tämä on täysin manuaalista työtä, jota haluamme välttää. Meidän piti siis rakentaa mekanismi, jossa käyttäjätunnukset poistetaan, kun niille määritelty päättymispäivämäärä on mennyt. Mekanismista halutaan tehdä automaattinen, jonka pitäisi helpottaa Usermgr-prosessin työmäärää.

Päätimme rakentaa käyttäjätunnusten vanhenemiselle mekanismin, joka on liitettynä jo aiemmin tehtyyn muutokseen, eli tietokantojen kiintiöihin. Käyttäjätietoihin lisäsimme arvot, joista käy selville käyttäjätunnusten aktivoitumis- ja päättymispäivämäärä, jonka avulla voimme kontrolloida käyttäjien oikea-aikaista poistumista järjestelmästä. Mekanismissa on Perl-ohjelmointikielellä tehty ohjelma, joka suoritetaan ajastetusti kerran päivässä. Ohjelma lähettää käyttäjälle ennakkovaroituksen kolme viikkoa ennen hänen tunnuksensa ja tietokantojensa sulkemista. Tällöin asiakkaalla on vielä hyvin aikaa reagoida, eli hän voi anoa lisäaikaa tietokannoilleen tai tehdä varmuuskopiot hallinnassaan olevista tietokannoista.

Käyttäjätunnusten vanhettua ne poistetaan, mutta ensin suoritetaan ylläpidollisia toimenpiteitä, joilla varmistetaan asiakkaiden tietojen säilyvyys. Asiakkaan tietokannoista otetaan määritetylle levyalueelle varmuuskopiot, jotta varmistutaan siitä, ettei kenenkään tietoja mene vahingossa hukkaan. Varmuuskopiot voidaan myös, asiakkaan niin halutessa, tuhota. Oletusarvoisesti tietokannoista otettuja varmuuskopioita säilytetään kuitenkin noin kolme kuukautta, jonka jälkeen ne poistetaan käytöstä.

7 PALVELUN ASIAKASKÄYTTÖÖN SAATTAMINEN

Teknisen puolen ollessa testausvaiheessa on myös itse asiakasliittymän järjestäminen ajankohtaista. Asiakkaiden liittäminen tietokantapalveluun järjestetään Usermgr-prosessin kautta, joka hallinnoi CSC:n käyttäjätunnuksia ja asiakkuuksia. Tarkoituksena on rakentaa jo olemassa olevan järjestelmän tyylinen käyttöliittymä, jolla voidaan hallinnoida asiakkaita tietokantapalvelussa. Haluamme automatisoida prosessin mahdollisimman pitkälle, jolla saamme vältettyä mahdollisia inhimillisiä virheitä. Virheitä sattuu erityisesti syöttövaiheessa, kun tietoja syötetään manuaalisesti järjestelmään sisään. Tällöin painottuu tietojen syöttäjän oma tarkkuus, koska esimerkiksi paperilla toimitettujen käyttölupahakemusten joukossa voi olla hyvinkin tulkittavissa olevia käsialoja.

Tietokantapalveluun tullaan hakemaan käyttölupaa erillisellä käyttölupalomakkeella, joka tulee asiakkaille saataville sähköisessä ja paperisessa muodossa. Käyttölupalomakkeessa määritellään tietokannalle erityinen vastuuhenkilö, joka on samalla myös kontaktihenkilö mahdollisissa ongelmatapauksissa. Tietokantapalvelun hakemukseen liittyy aina CSC:n ympäristöön avattu laskentaprojekti, koska tämä helpottaa asiakkuuksien hallinnoinnissa. Jos asiakas kuuluu useampaan kuin yhteen laskentaprojektin, on hänen täsmennettävä, mihin projektiin tietokantapalvelua avataan. Mikäli asiakkaalla ei ole lainkaan laskentaprojektia CSC:llä, on hänen ensin hankittava laskentapalveluiden käyttölupa.

Käyttölupalomakkeen tekstin ja sanamuotojen muokkaaminen ja työstö kesti yllättävän kauan. Saimme apua Asiakasprosessit ja –ratkaisut ryhmältä, joka käsittelee uusia ja vanhoja asiakaskontakteja. Heille vastuualueelleen kuuluu myös uusien sopimusten muokkaus, jos se koskee uutta käyttöön tulevaa palvelua. Tehtyjä muutoksia ei voitu ottaa käyttöön välittömästi, koska ne täytyi hyväksyttää palvelualueen johtoryhmän kokouksessa. Kokouksessa käyttölupalomake ei mennyt heti läpi ja siihen jouduttiin tekemään muutamia johtoryhmän toivomia muutoksia, joka tietenkin venytti koko projektin kestoa.

Avattavan tietokannan ja sen tunnusten päättymispäivä asetetaan samaksi kuin siihen liittyvän laskentaprojektin päättymispäivä. Tietokantapalvelun kesto on siten kuitenkin enintään kaksi vuotta sen avaamispäivästä. Asiakkaan toiveesta voidaan tietokantapalvelun kesto rajoittaa lyhyemmäksi. Jos asiakas haluaa uusia tietokantapalvelun sopimuksensa, on hänen täytettävä tietokantapalvelun uusintahakemus.

Tietokantapalvelun ohjeet tehtiin projektin sivutuotteena. Ohjeisiin panostettiin voimavaroja rajoitetusti, koska oletamme että palvelua käyttävät osaavat jo tietokantaoperoinnin perusteet. Ohjeissa keskityttiinkin lähinnä CSC:n oman ympäristön mahdollisiin kompastuskiviin ja miten ne voidaan välttää. Ohjeet sisältävät paljon esimerkkejä, jotka auttavat käyttäjiä hah-

mottamaan tärkeimpiä toimintoja, joita he voivat käyttää. Ehkä tärkeimpänä kohtana ohjeissa nähtiin kohta, jossa tietoja siirretään tietokantaan ja miten niitä sieltä saa ulos. (CSC 2009.)

8 PÄÄTELMÄT JA PROJEKTIN ONNISTUMINEN

Projekti oli toteutuksen ja projektin johtamisen kannalta mielestäni hyvin onnistunut. Projektilla oli selkeä päämäärä, tarvittavat resurssit ja riittävä aika toteutukseen. Osatehtävät olivat projektisuunnitelmissa mielestäni realistisesti suunniteltu ajankäytön suhteen. Suuria heittoja sen suhteen ei havaittu ja projekti eteni aikataulussaan aina käyttöönottoon asti, jolloin aikataulusta jouduttiin poikkeamaan. Poikkeaman aiheutti oma optimistinen arviomme käyttöönoton työmäärästä, joka yllätti meidät kaikki. Käyttöönoton hallinnollinen puoli, johon kuului esimerkiksi käyttölupapolitiikan määrittämisen, vei todella paljon aikaa projektin loppuvaiheessa. Projekti olikin teknisessä mielessä valmiina jo useampia kuukausia, mutta hallinnollisen puolen ollessa jäissä, jouduimme lykkäämään tietokantapalvelun virallista käyttöönottoa muutamaan kertaan.

Projekti ei edetessään kohdannut suuria vaikeuksia. Hitain vaihe oli teknisen toteutuksen osalta palvelinten testausvaihe, jossa jouduttiin asentamaan käyttöjärjestelmiä, ohjelmistoja ja tietenkin itse tietokantaohjelmisto. Tähän kului paljon työaikaa, mutta halusimme tehdä testaustyön perusteellisesti, josta projektin loppuvaiheessa hyödyimme. Teimme, ainakin omasta mielestäni, täysin oikeat valinnat palvelimen suhteen. Käyttöjärjestelmään emme voineet juurikaan vaikuttaa, mutta toistaiseksi olemme olleet tyytyväisiä nykyiseen järjestelmään. Toiminta on ollut kaikin puolin luotettavaa, eikä virhetilanteita ole esiintynyt. Muutama itseaiheutettu katko on sattunut, mutta tästä voimme syyttää vain omaa tietämättömyyttämme.

Projektiryhmä oli motivoitunut viemään projektin kunnialla läpi. Kokoonnuimme viikoittain projektikokoukseen, jossa käytiin läpi edellisellä viikolla suoritetut tehtävät ja niissä mahdollisesti ilmenneet ongelmat. Kokouksessa käytiin läpi myös tulevia tehtäväkokonaisuuksia, niiden tärkeysjärjestystä ja suoritusaikatauluja. Järjestelmällisillä kokouksilla saimme organisoitua tehtävät hyvin, eikä suuria heittoja tämän ansiosta asetettuun aikatauluun tullutkaan.

Asetetut päämäärät projektille olivat tärkeitä. Kukin projektin osallistuja tiesi tarkalleen mitä oltiin tekemässä ja minkälaisia lopputuloksia oltiin hakemassa. Oli myös motivoivaa tietää, että luotua palvelua tullaan käyttämään asiakaskäytössä, joten työn toteutuksessa oli syytä olla tarkkana. Asiakkaamme ovat hyvin laatutietoisia ja laadulliset ongelmat tulevat hyvin nopeasti esille.

Opimme todella paljon työskennellessämme projektissa. Löysimme uusia tapoja ratkaista ongelmia ja miten välttää mahdollisia ongelmia. Saimme paljon arvokasta tietoa eri laitteistokokoonpanojen toiminnasta ja niiden yhteensopivuudesta tietokantakäyttöön. Dbadm-

prosessiin kuuluvien henkilöiden omat tiedot lisääntyivät projektin aikana todella paljon ja itse sain uutta tietoa suurlaskentaympäristöstämme.

Virallisesti projekti päätettiin jo syystalvella 2008, mutta varsinainen käyttöönotto viivästyi. Ongelmia käyttöönotossa aiheutti lähinnä hallinnolliset yksityiskohdat, joita ei osattu projektisuunnitelmassa huomioida. Tietokantapalvelu avattiin virallisesti asiakaskäyttöön 23.3.2009.

Lähteet

Kirjallisuusviitteet

Gilfillan, I. 2003. Mastering MySQL 4. Alameda: SYBEX

van der Lans, R. 1988. Introduction to SQL. Kent: Addison-Wesley Publishers Ltd.

Manis, R., Schaffer. & Jorgensen R., 1988. UNIX Relational Database Management. Englewood Cliffs: Prentice-Hall

Wall, L., Christiansen T. & Schwartz R., 1996. Programming Perl . 2. painos. Sebastopol: O'Reilly

Zawodny J. & Balling D.,2004. High Performance MySQL – Optimization, Backups, Replication & Load Balancing. Sebastopol: O'Reilly

Sähköiset lähteet

CSC – Tieteen Tietotekniikan Keskus. 2009a. Datan tallennus. Viitattu 12.03.2009 http://www.csc.fi/english/research/datastorage/index_html

CSC – Tieteen Tietotekniikan Keskus. 2008. CSC:n historia. Viitattu 11.11.2008 http://www.csc.fi/csc/csc/historia/index_html

CSC – Tieteen Tietotekniikan Keskus. 2009b. Tietokantapalvelun ohje. Viitatu 15.04.2009 http://www.csc.fi/tutkimus/tallennus/tietokantapalvelu/tietokanta_kaytto

It-viikko. 2009. Myös Mårten Mickos hyvästelee Sunin. Viitattu 15.04.2009 http://www.itviikko.fi/ihmiset-ja-ura/2009/02/08/myos-marten-mickos-hyvastelee-sunin/20093460/7

MPoli Oy Ext3. 2008a. Linux-käyttöjärjestelmän Ext3-tiedostojärjestelmän tekniikka. Viitattu 15.03.2009. http://linux.fi/index.php/Ext3

MPoli Oy XFS. 2008b. Linux-käyttöjärjestelmän XFS-tiedostojärjestelmän tekniikka. Viitattu 15.03.2009. http://en.wikipedia.org/wiki/XFS

Redhat Inc. 2008. Satellite-ylläpitopalvelimen tekninen asennusohje. Viitattu 10.02.2009 http://www.redhat.com/docs/manuals/satellite/Red_Hat_Network_Satellite-5.1.1/html/Installation_Guide/ch-intro.html

Sun Microsystems Inc. 2008. Lehdistötiedote MySQL:n siirtymisestä Sunille. Viitattu 10.02.2009 http://www.mysql.com/news-and-events/sun-to-acquire-mysql.html

Wikipedia. 2008. MySQL:n käytöstä. Viitattu 13.04.2009. http://en.wikipedia.org/wiki/Mysql

Kuvaotsikkoluettelo

Taulukko-otsikkoluettelo